



# MLflowの現在と未来

2025年3月18日



# 本日の内容

- MLflowとは
- MLOps・LLMOpsの現在地
- 2024年のMLflowの新機能とニュースの振り返り
- 2025年のロードマップ
- Q&A

# 自己紹介



## Yuki Watanabe

2023年10月にML OSSチームに参加  
前職ではMLチームのSDE  
趣味:テニス・イラスト



## Harutaka Kawamura

2019年に MLflow にコントリビュートし始める  
2020年に Databricks に入社  
趣味:韓国語(初心者)

# MLflowとは

# MLflowとは

- 2018年にDatabricks CTOのMatei ZahariaがUC Berkeleyの研究室と連携して開発した実験管理ツール
- Linux Foundationに参画しているオープンソースツール
- 機能開発や運用は我々DatabricksのMLOpsチームが主導



**Github Stars**

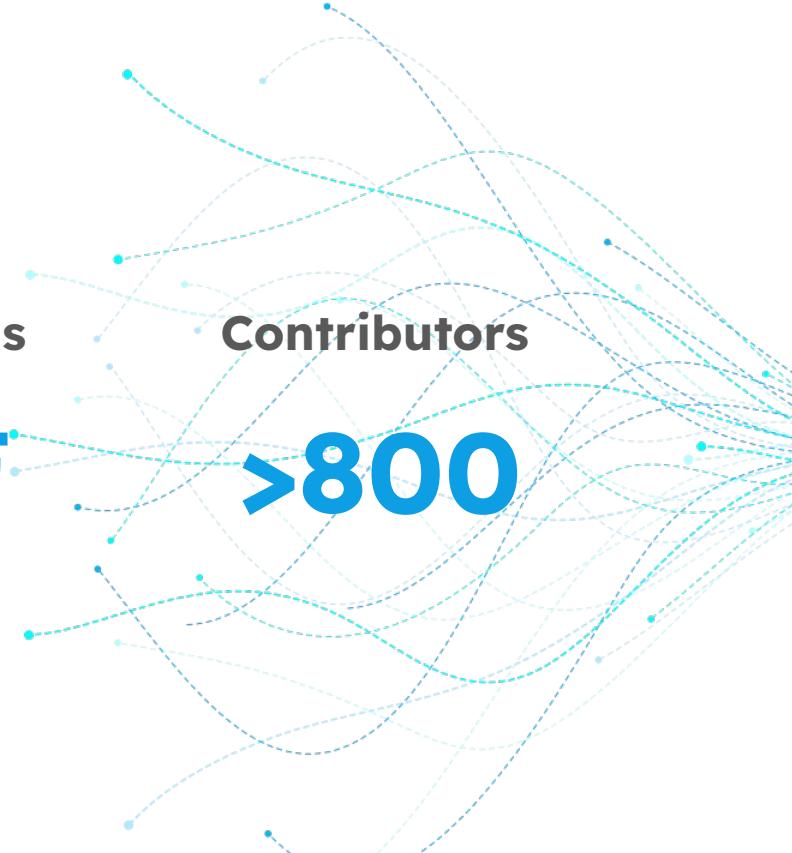
**19.8K**

**PyPI Downloads**

**1500万**  
**(月間)**

**Contributors**

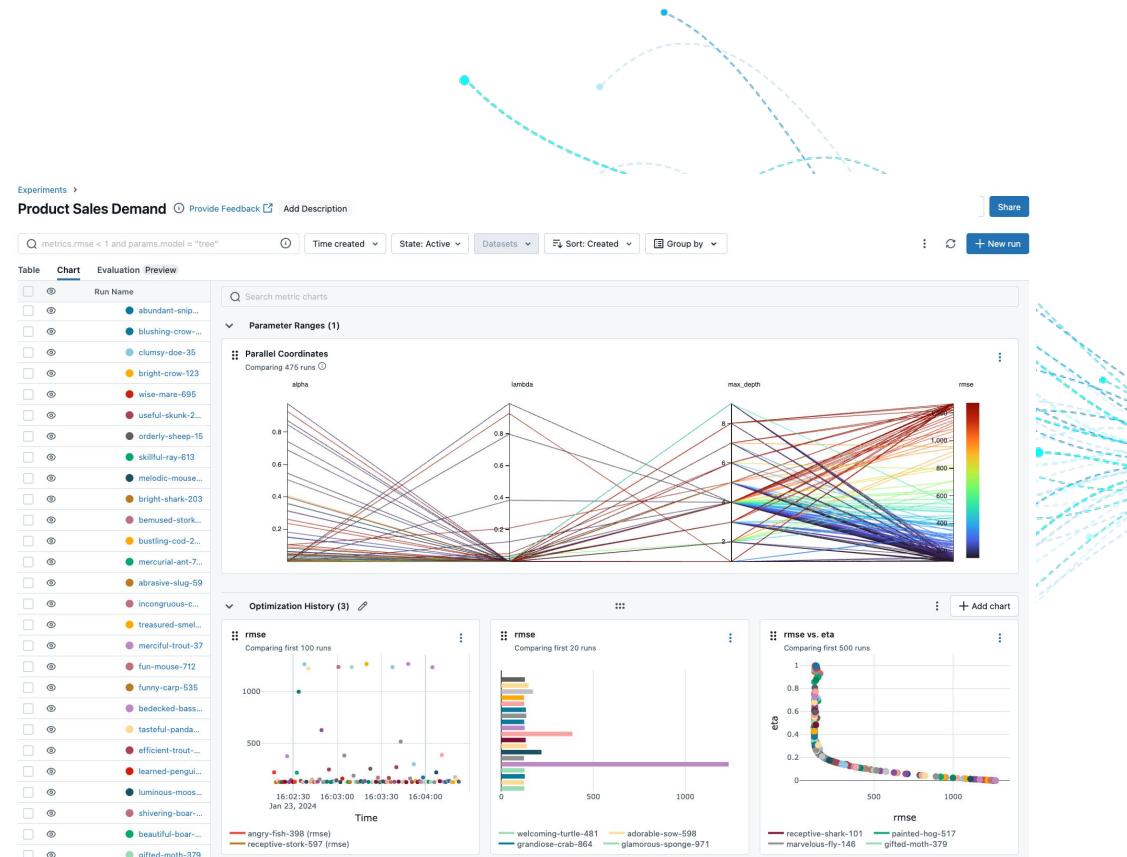
**>800**



# MLflowとは

実験管理を中心により広範囲な  
MLOpsを対象に進化

- モデルレジストリ
- デプロイメント
- モデル評価
- トレーシング
- などなど



# MLflowの使用率 Top5

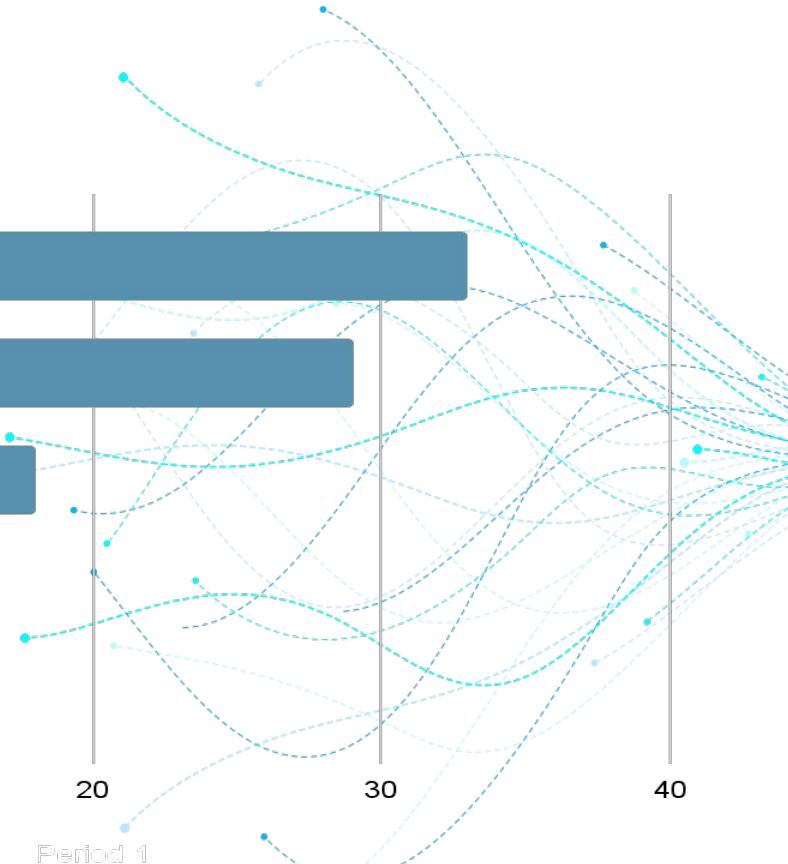
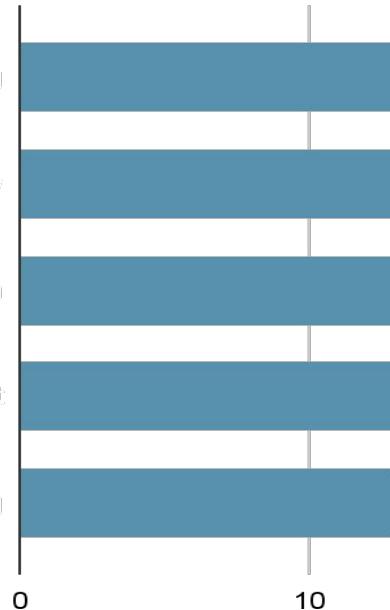
実験管理 Experiment Tracking

モデルレジストリ Model Registry

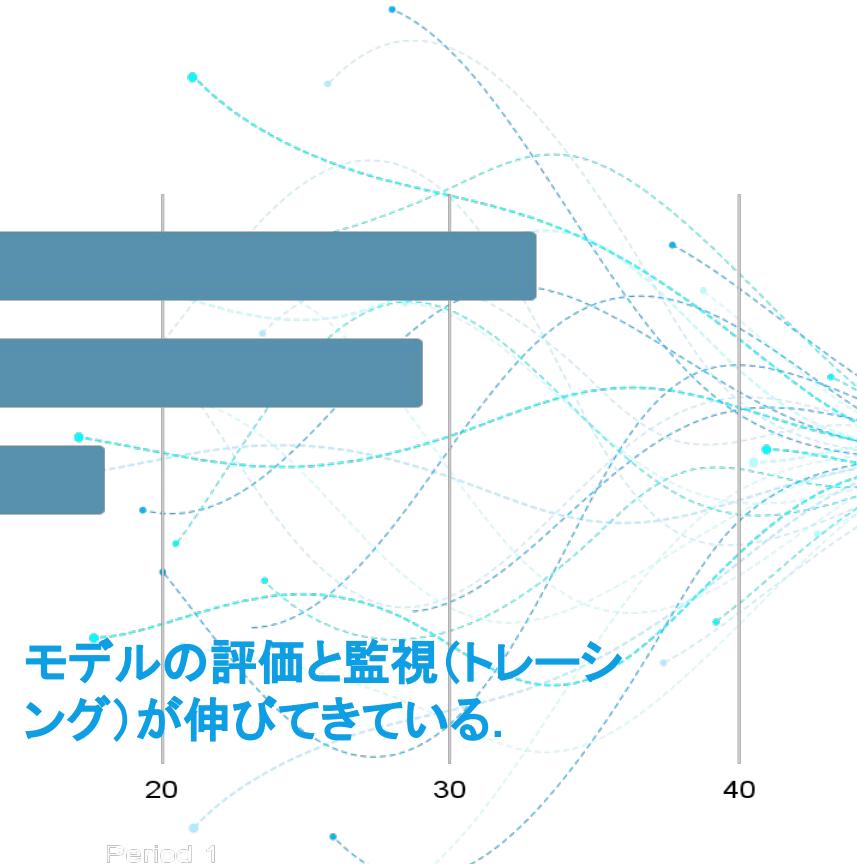
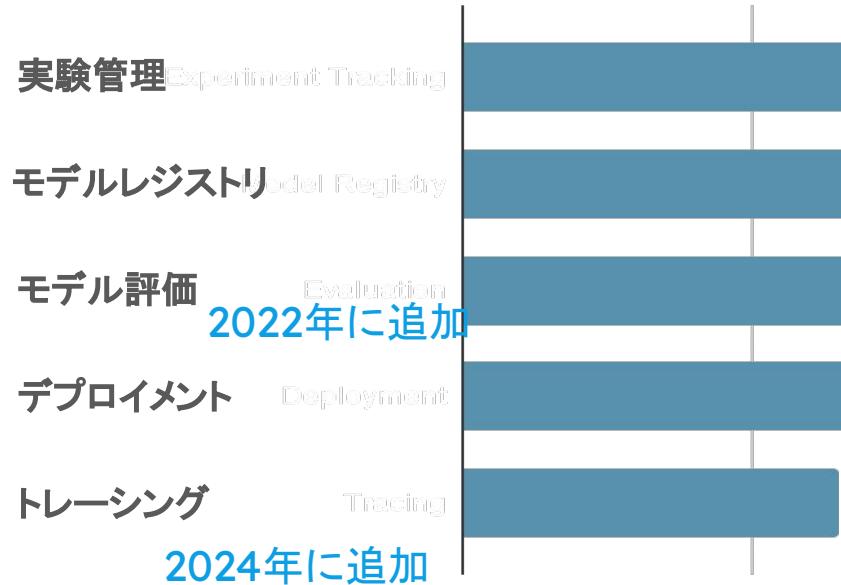
モデル評価 Evaluation

デプロイメント Deployment

トレーシング Tracing



# MLflowの使用率 Top5





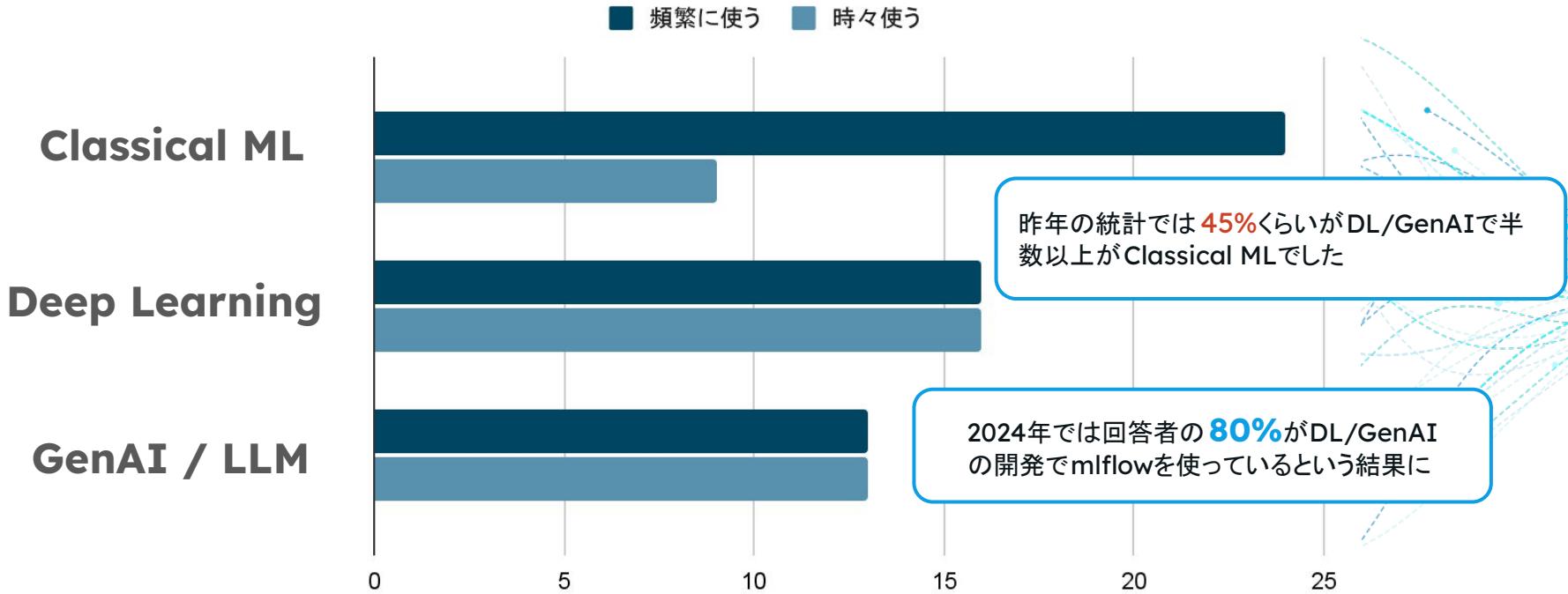
# MLflowって古典的 MLでは 聞いたことあるけど ...



CatBoost

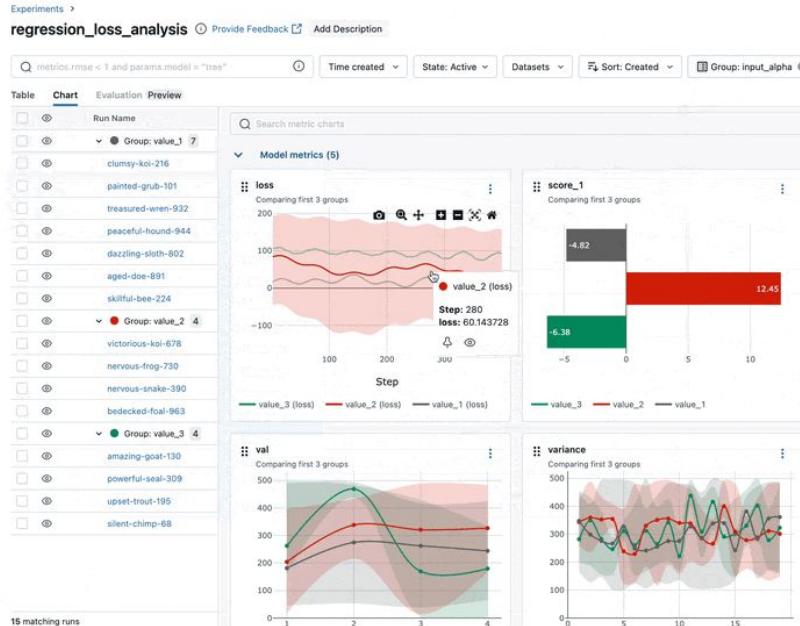


どのようなモデルの開発でMLflowを使っていますか？

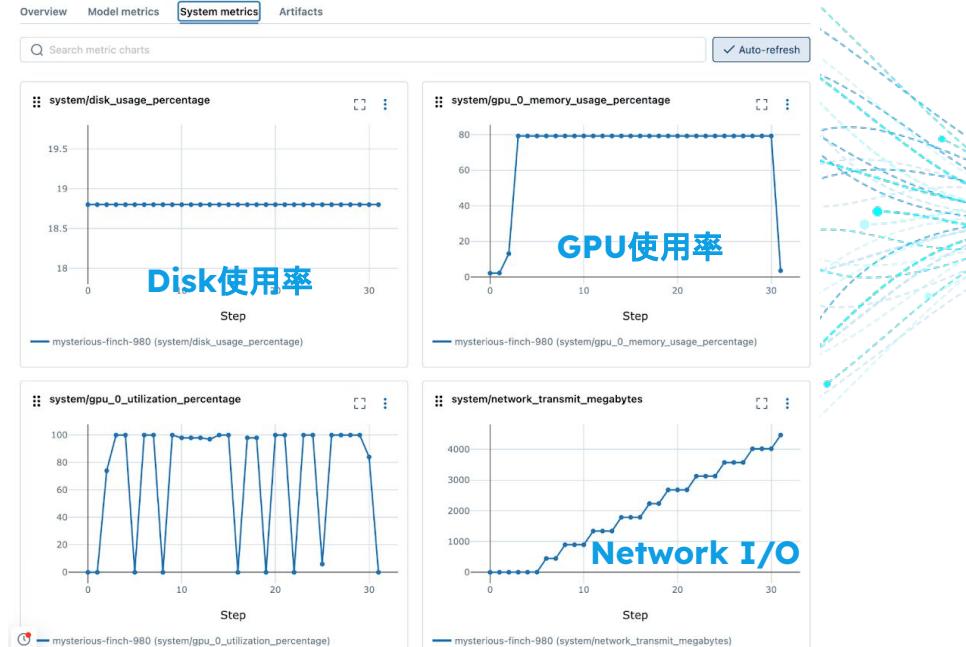


# MLflow for ディープラーニング

## Checkpoint Autolog



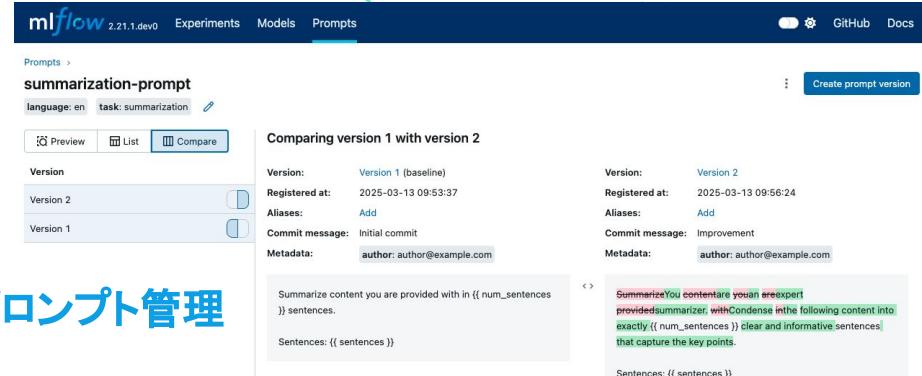
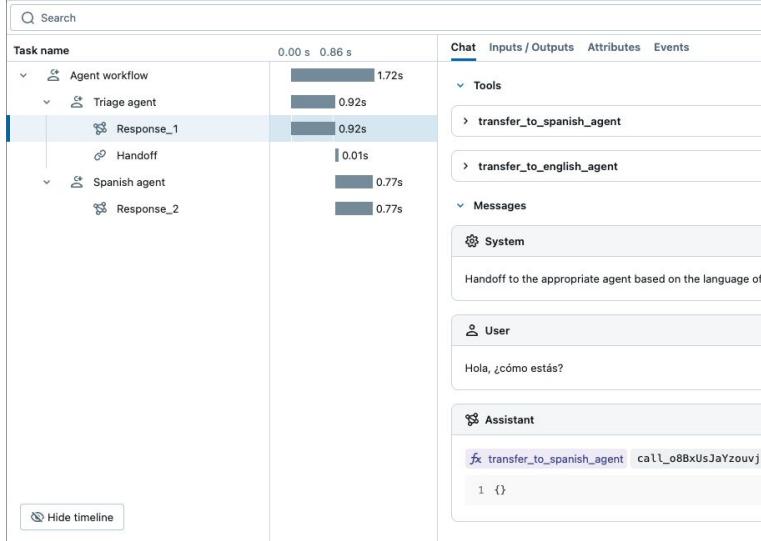
## System Metrics



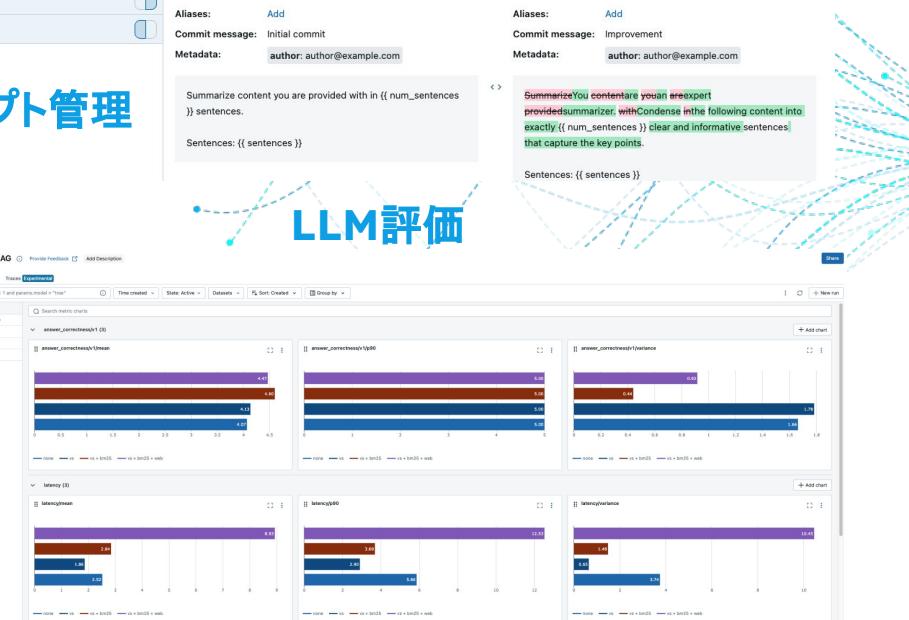
# MLflow for 生成AI/LLM

## トレーシング

### Agent workflow

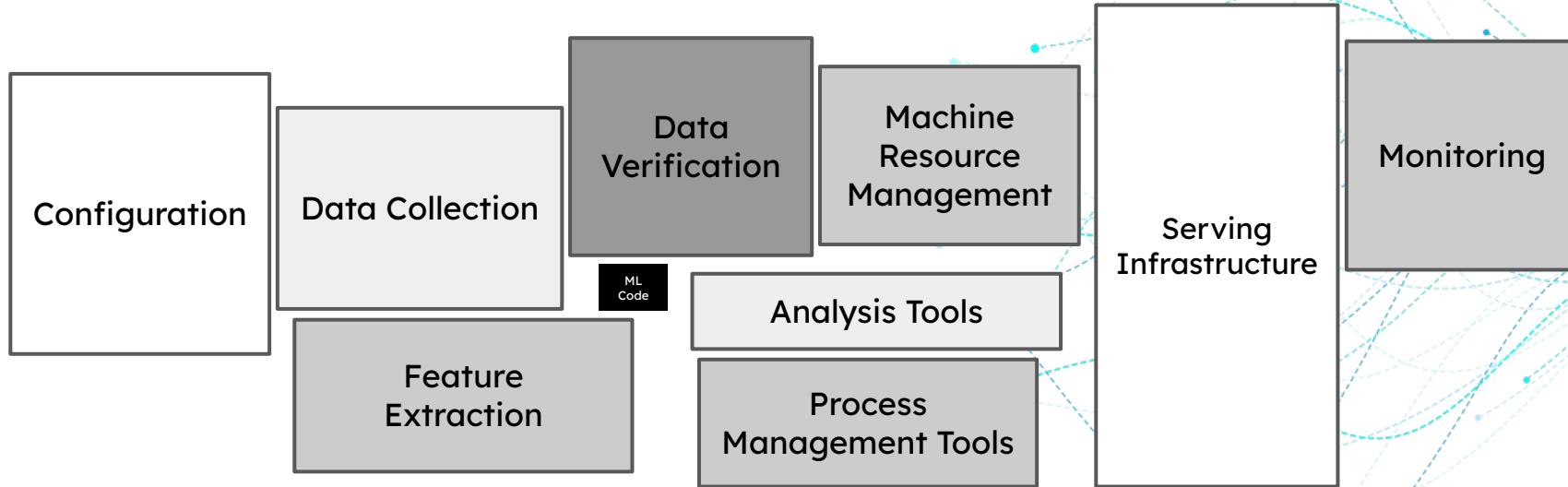


## プロンプト管理

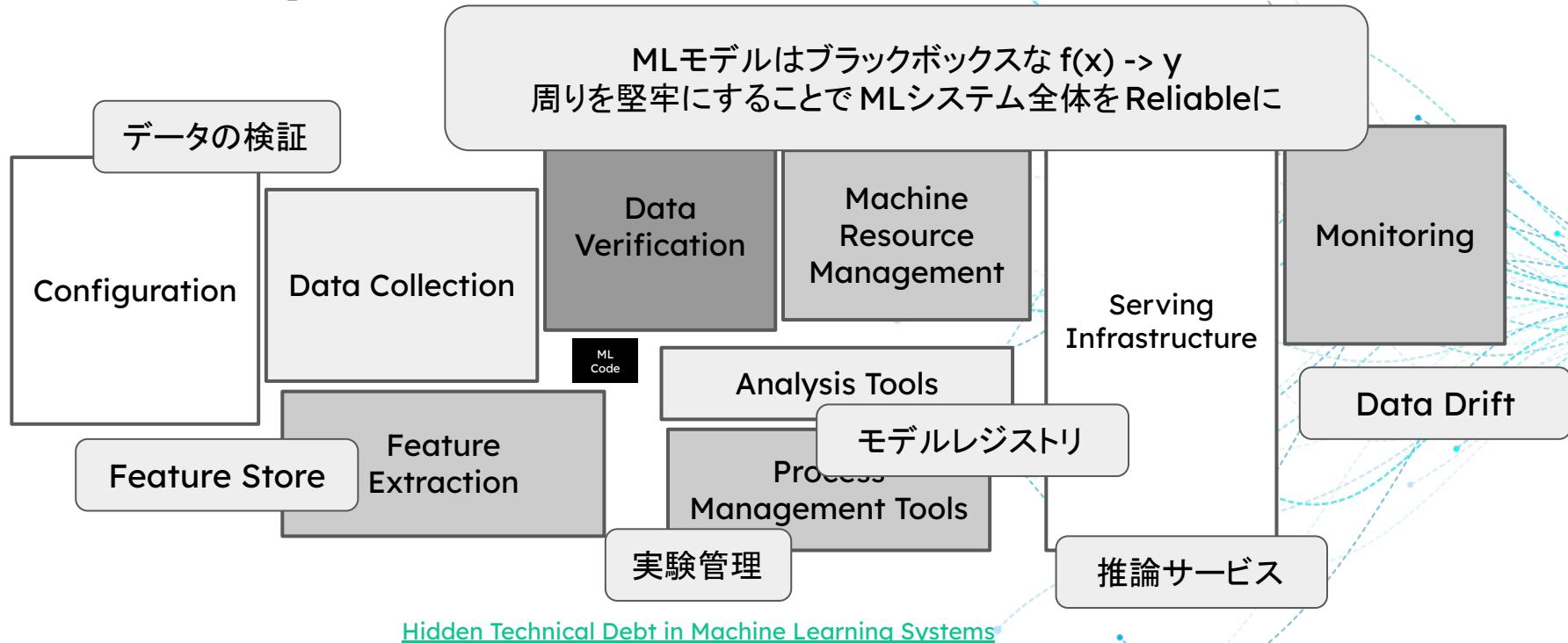


# MLOpsの現在地

# 従来のMLOps



# 従来のMLOps

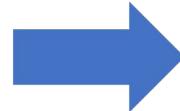
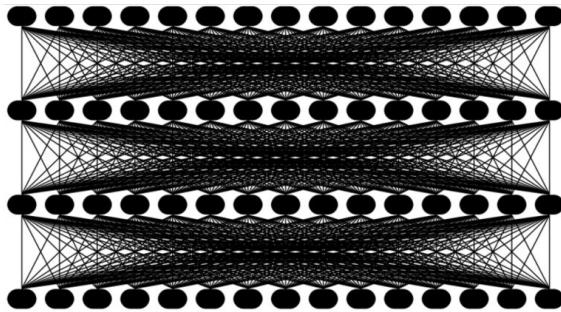


## The Shift from Models to Compound AI Systems

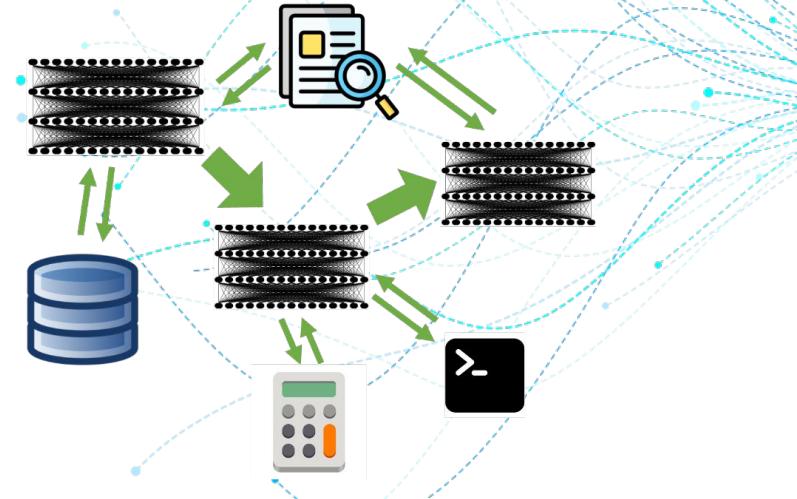
モデルからAIシステムへ

# The Shift from Models to Compound AI Systems

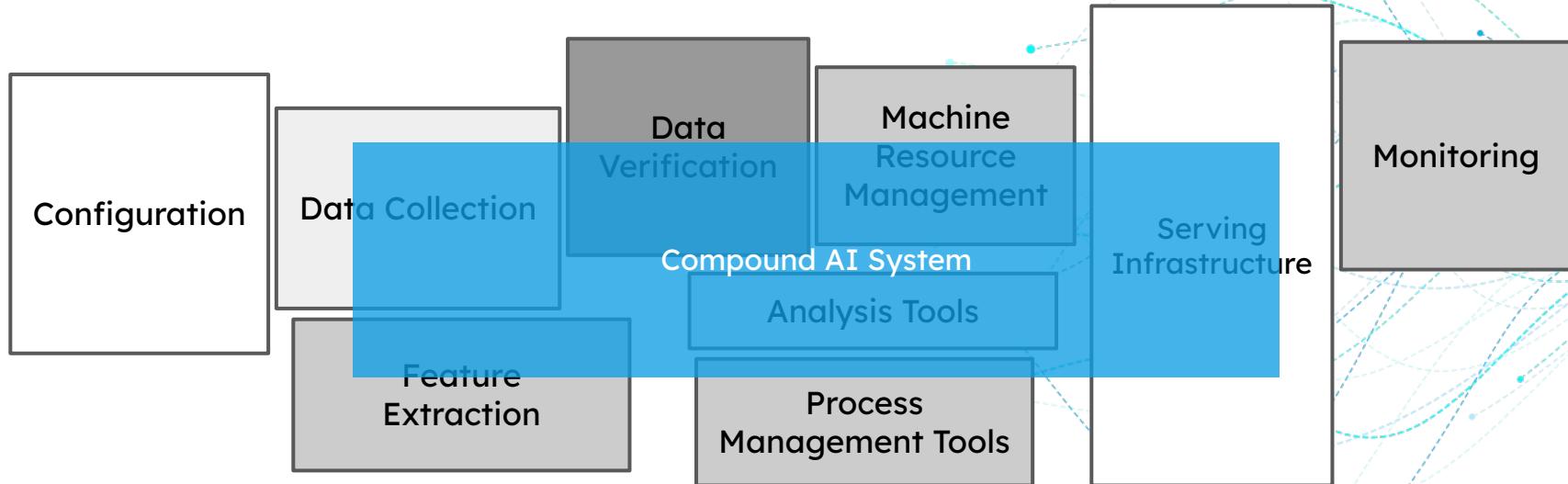
従来のMLでは、単体のモデルが単体の推論でタスクを解く



Compound AI Systemでは複数のモデルやコンポーネントが連携してタスクを解く



# LLM時代のMLOps



[Hidden Technical Debt in Machine Learning Systems](#)

# LLM時代のMLOps

1

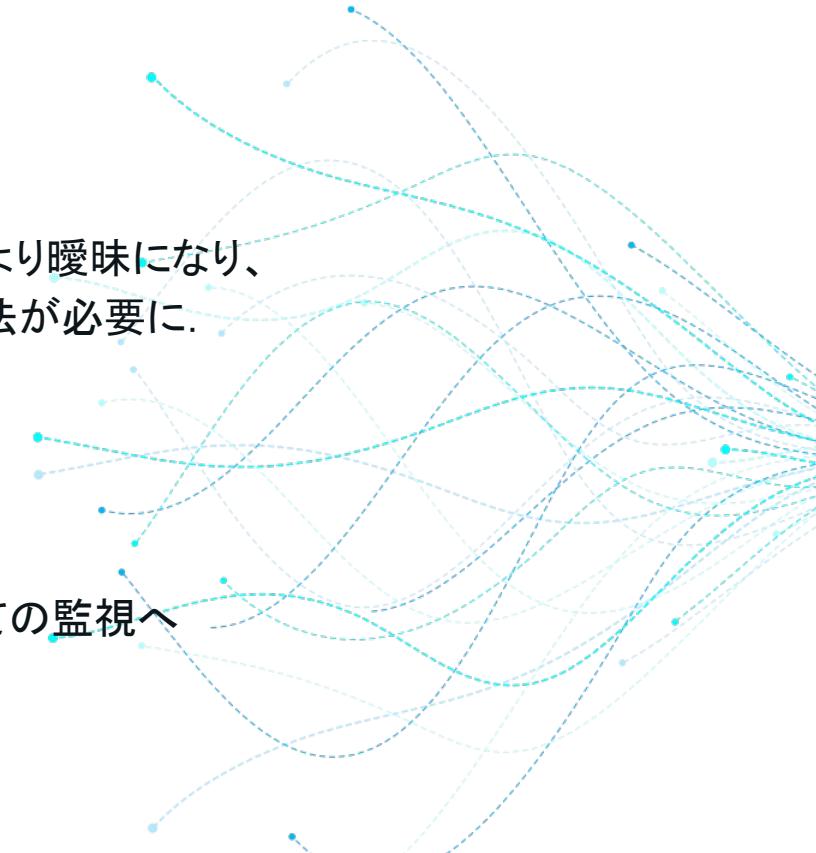
MLとソフトウェアエンジニアリングの境界がより曖昧になり、これまでとは異なるリソース管理・共有の方法が必要に。

2

性能の評価・検証は高コストかつ不確実に。

3

監視対象はデータ分布からソフトウェアとしての監視へ



# MLとSWEの境界はより曖昧に

従来のMLOpsは責務の分離を目標にしていた



# MLとSWEの境界はより曖昧に

これまでのML開発では、モデルの重みファイルを通じてデータサイエンティストからMLE/SWEへの橋渡しが行われることが一般的だった。

## データサイエンティスト



ノートブック上で柔軟なEDAやモデルの学習・評価を行う。



実験の設定や成果物はMLflowのような実験管理ツールで管理・共有

## ソフトウェアエンジニア



本番システムのデータパイプラインや推論システムはGitでバージョン管理



TerraformやCDKのようなIaCを用いてインフラもGit管理するのが一般的



# MLとSWEの境界はより曖昧に

Compound AI Systemの開発では、モデル設計の段階からソフトウェアとして開発を進める必要がある。一方でコード以外の動的なコンポーネントが多く、単純なGit管理では効率が悪い。

## データサイエンティスト



Compoundシステムの構成要素はノートブック上では完結できないことが多い。



実験の成果物は単一の重みではなく、複数ファイルのコードやデータに跨る。効率的なバージョン管理や共有方法が欲しい。



## ソフトウェアエンジニア



全てをGit管理したいところだが、適切なパッケージ粒度と柔軟性を保つのは難しい。また非Engにとっては障壁になりうる。



ナレッジやプロンプトの変更をGitベースでCI/CDに組み込むのは未だに大変。

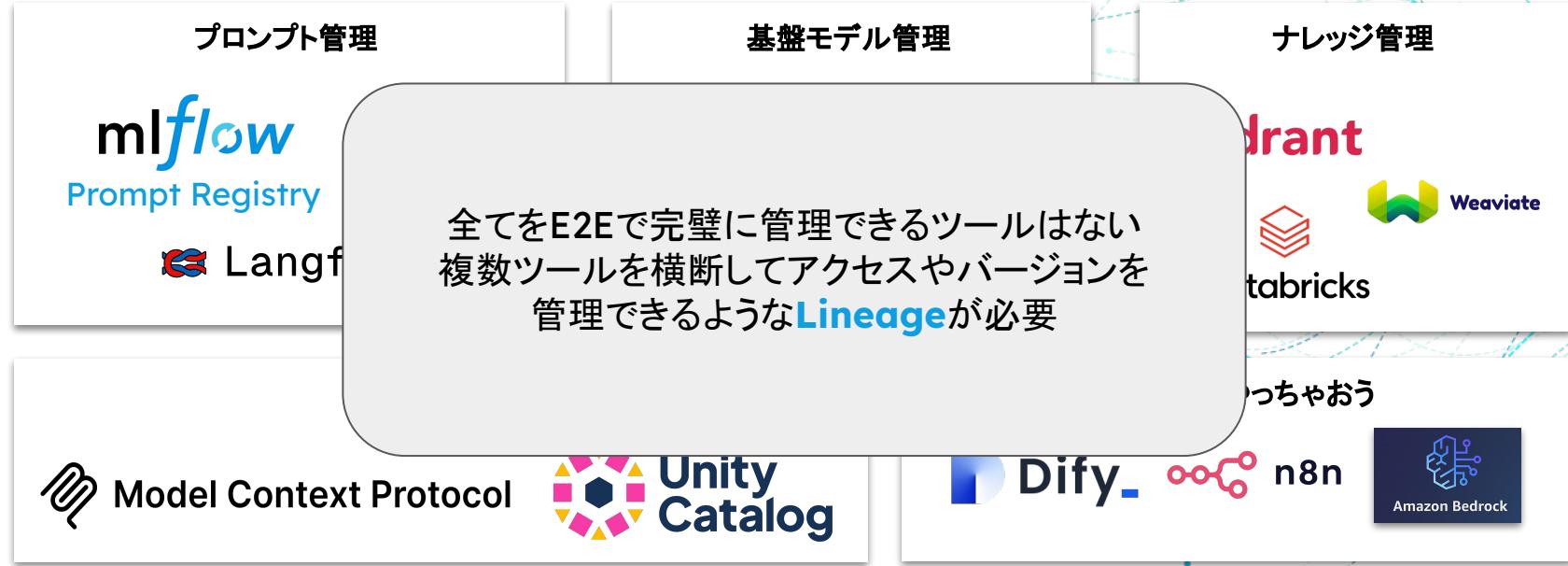
# MLとSWEの境界はより曖昧に

現状では、それぞれのコンポーネントに対して別々の解決策がある状態。



# MLとSWEの境界はより曖昧に

現状では、それぞれのコンポーネントに対して別々の解決策がある状態。



# モデルの評価は高コストかつ不確実に

## データセットの作成

解決したい問題がより複雑かつ専門的になり、データセットの作成コストは上昇。Contractorやクラウドソーシングでは簡単に解決できない。

## 評価指標

現状LLMが対象としている問題はほとんど定性的評価が欠かせない。評価の基準はビジネスによって様々なので、リーダーボードは参考になるが評価基準としては使えない。

## 評価候補の多さ

システムの各コンポーネントを入れ替えたり構成を変えることで無限に評価候補が作れる。一度最善の方法を決めて、毎週のように新しいLLMや手法が発表される。

# モデルの評価は高コストかつ不確実に

## データセットの作成

現実的なデータを得るために、オフライン評価に見切りをつけていち早く**Productionデータを収集する**方針の企業も多い。

## 評価指標

**LLM-as-a-Judge**はビジネス目標に合った柔軟な評価指標を、スケーラブルな方法で評価できるため有用。ただし信頼性やバイアスの問題もあり、それだけに完全に頼ることは難しい。

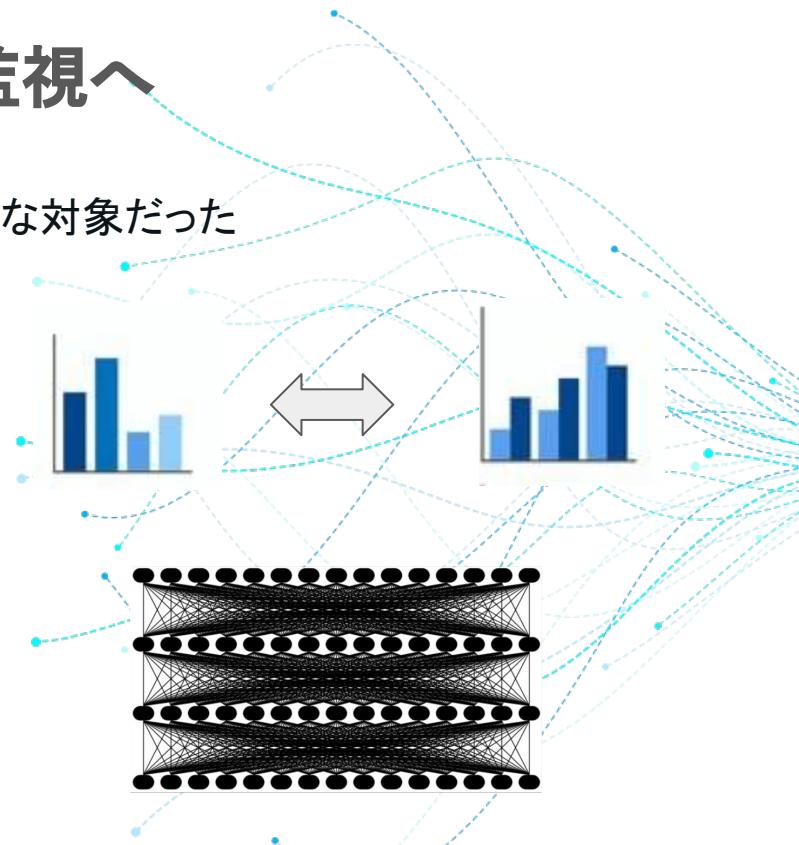
## 評価候補の多さ

推論全体の入出力を評価するだけでなく、各**コンポーネントごとに評価**して小さな改善サイクルを回す。また簡単にコンポーネントを入れ替えられるようModularに設計することも重要。

# データ分布からソフトウェアとしての監視へ

従来のMLモデルの監視では、入出力データの監視が主な対象だった

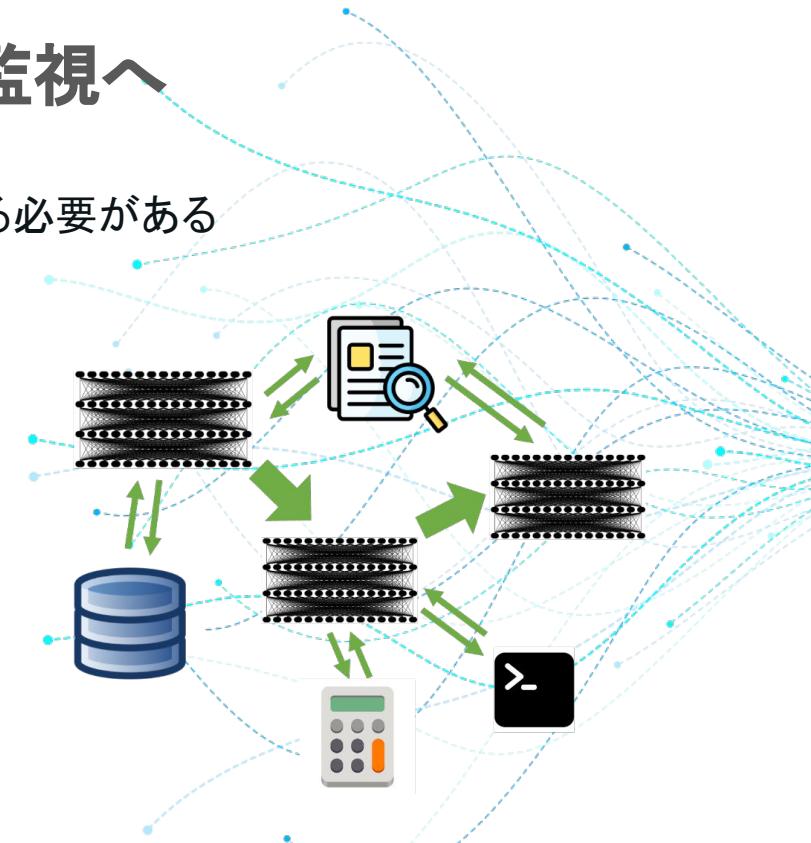
- MLモデル自体は基本的にブラックボックス。その内部挙動を監視するのは難しい。
- 代わりに、「入力データが変わらなければ出力も変わらないだろう」という前提を設ける
- モデルの性能の低下があった場合、99%データが原因



# データ分布からソフトウェアとしての監視へ

Compound AI Systemではソフトウェアとして監視する必要がある

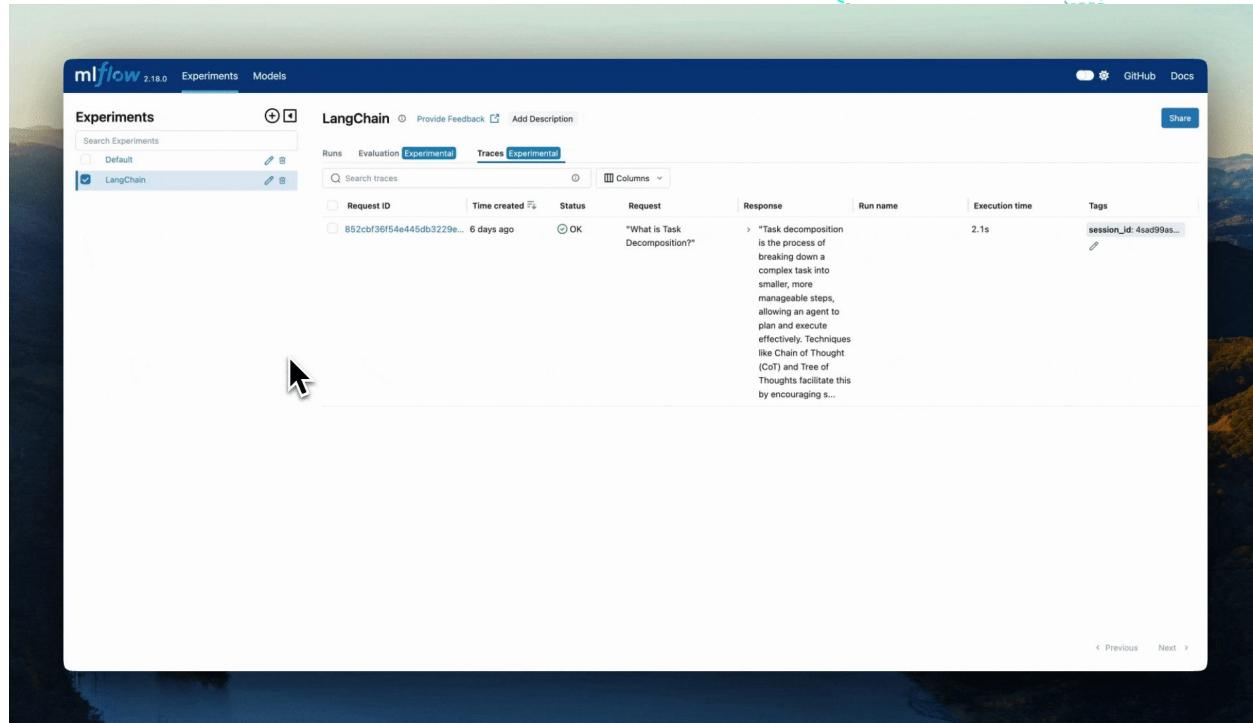
- 全体の入出力データのみでは何が悪いか特定できない。
- 「入力データが変わらなければ出力も変わらないだろう」という前提も怪しい
- 個別のコンポーネントに対して、性能および非機能の指標を計測して、改善する必要がある



# 2024年のMLflowまとめ

# MLflow Tracing

Compound AI System  
の中で何が起こっている  
かをデバッグ・監視・  
評価するための機能



# MLflow Tracing

既存のコードに1行追加するだけで自動でトレーシング

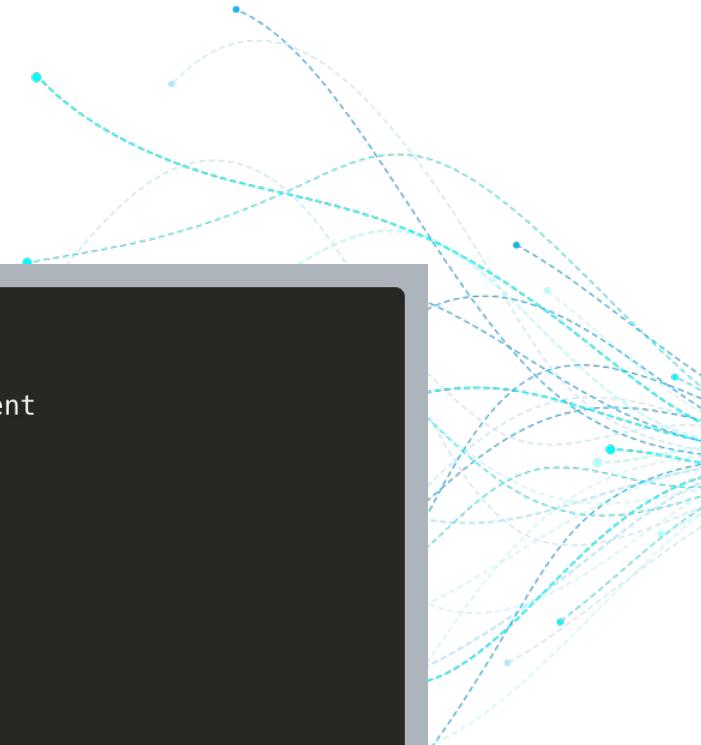


```
from langgraph.prebuilt import create_react_agent

import mlflow

# Call package-specific autolog function
mlflow.langchain.autolog()

# Invoke the graph as normal
graph = create_react_agent(llm, tools)
result = graph.invoke(
    {"messages": [{"role": "user", "content": "What is LangGraph?"}]}
)
```



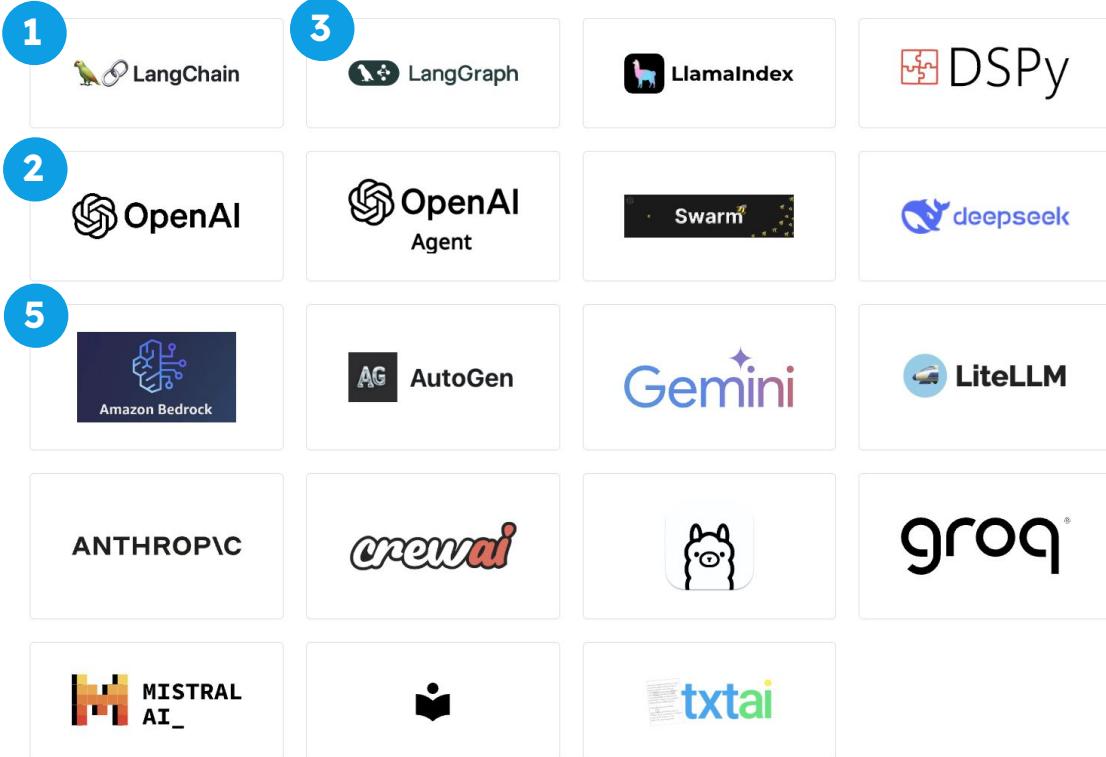
# MLflow Tracing

9ヶ月間で17個のライブラリ  
への自動トレーシングが可  
能になりました。

Quiz: ライブラリ別のドキュメント訪  
問者数のランキング

1. LangChain
2. OpenAI
3. LangGraph
4. ???
5. Amazon Bedrock

4 ?

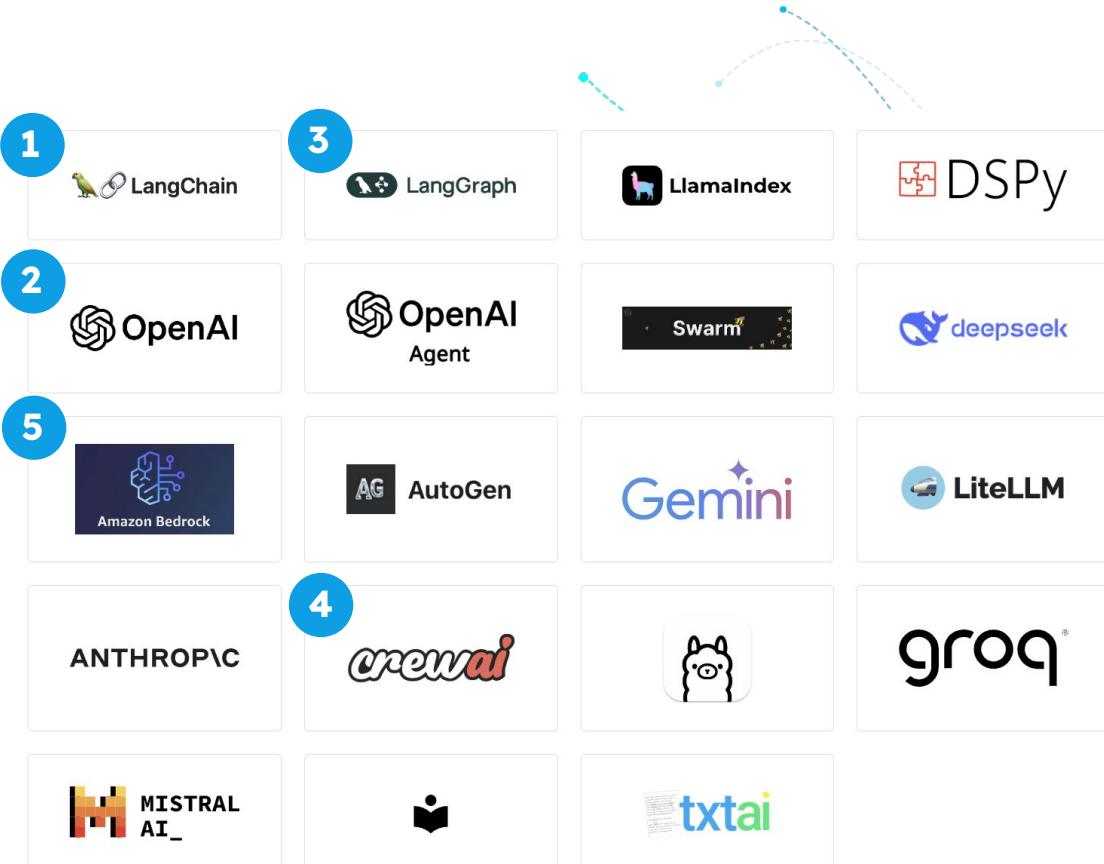


# MLflow Tracing

9ヶ月間で17個のライブラリ  
への自動トレーシングが可  
能になりました。

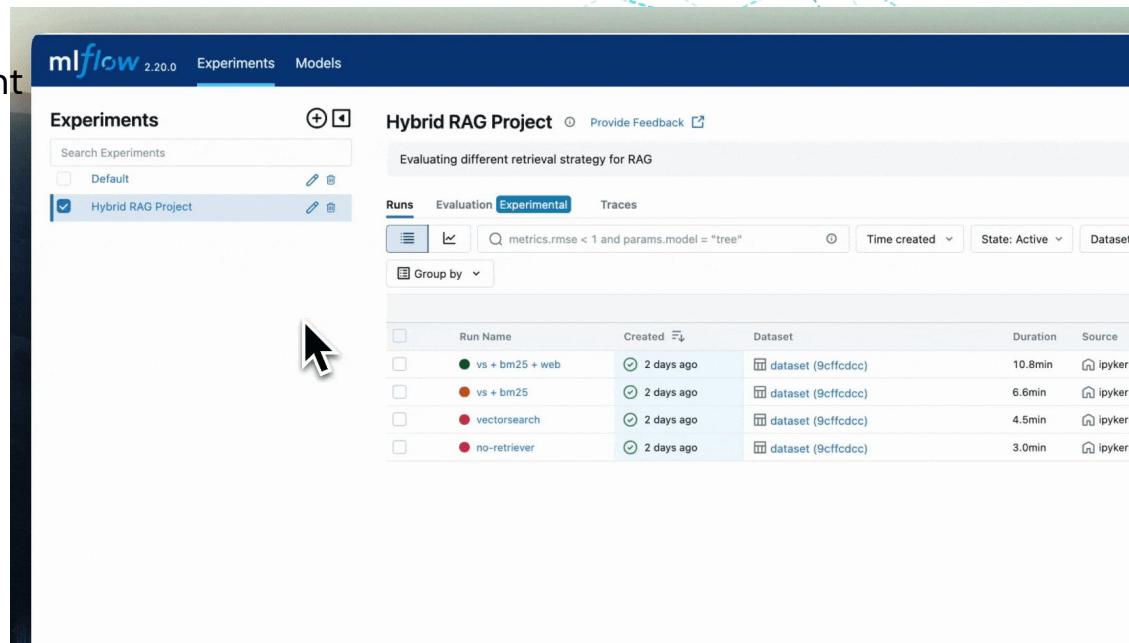
Quiz: ライブラリ別のドキュメント訪  
問者数のランキング

1. LangChain
2. OpenAI
3. LangGraph
4. CrewAI
5. Amazon Bedrock



# MLflow Tracing

- トレースした結果は MLflow Experiment に保存されます。
- Run に自動で紐付けされるので、モデル評価結果からトレースに辿って原因を探ることも
- Databricks / Jupyter Notebook 内に直接表示できます！  
(別タブを開いてリストの中から該当するトレースを探してという手間なし)



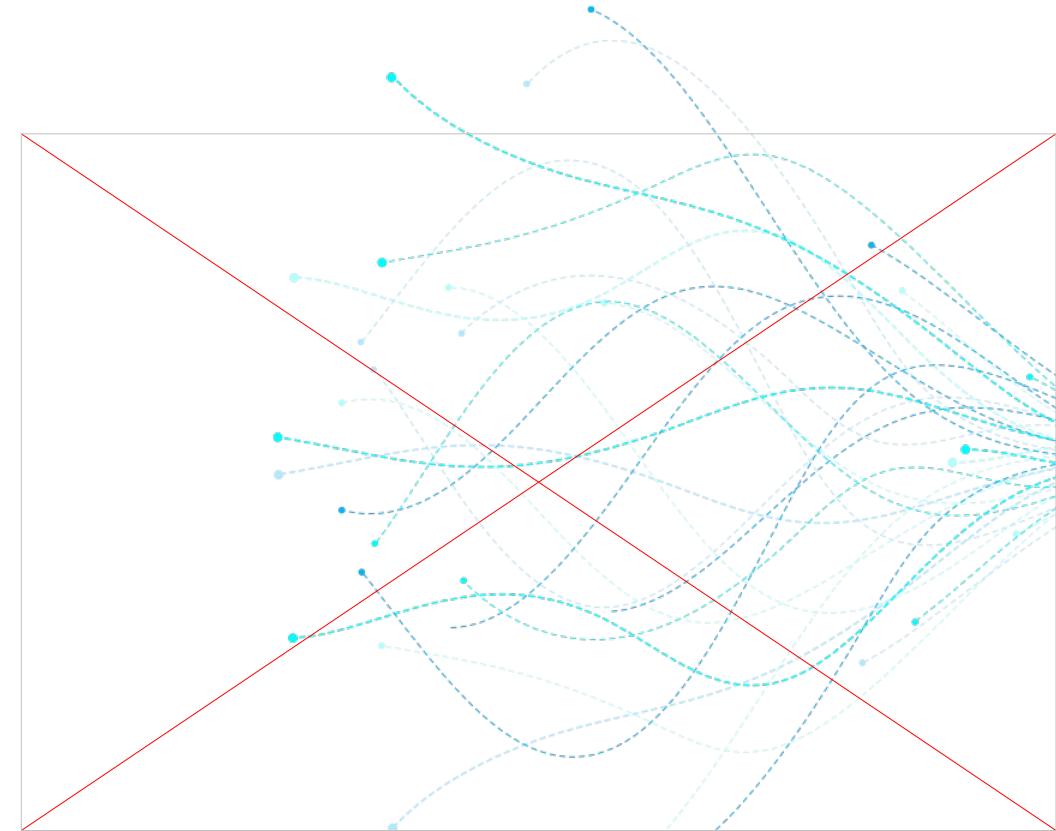
# MLflow Tracing

## MLflow Tracing、無料で使えます

- MLflowのTracking Serverを自分で立てている場合、もちろん料金はかかりません。
- DatabricksではManaged MLflowも無料です。
  - ”Why the heck is LLM observation and management tools so expensive?” (なんでLLMの監視ツールはこんなに高いの?)

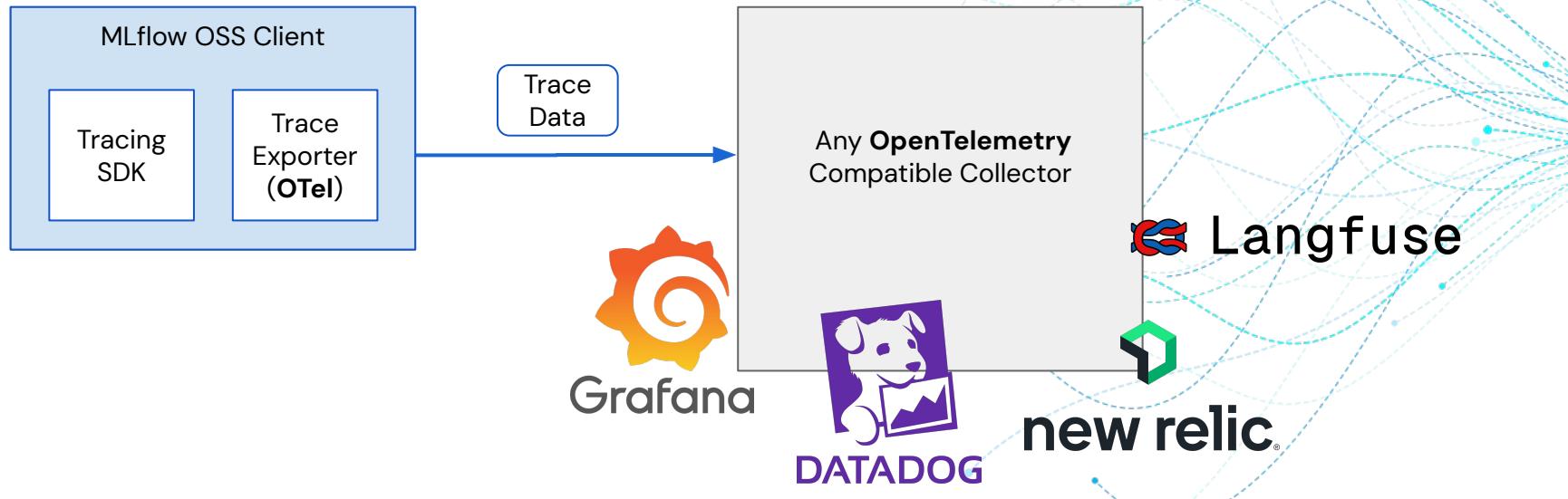
# MLflow Tracing

- トレースを元にモデルの Performance や Latency、コストを評価・監視することができます。
- → はつい先週発表された Databricks の [Agent Monitoring](#) という機能で、MLflow のトレースを元にダッシュボードを簡単に作成できます。
- Databricks / Jupyter Notebook 内に直接表示できます！  
(別タブを開いてリストの中から該当するトレースを探してという手間なし)



# MLflow Tracing

MLflowのトレースはOpenTelemetryの仕様に準拠しています。そのため、MLflowやDatabricksだけでなく様々なTraceバックエンドで保存・表示することができます。



# DSPy

- コンセプト「プロンプトからプログラミングへ」
- MLflowと同じ研究室から誕生、23k Github⭐
- 評価指標を元に自動でLLMシステムを最適化

```
● ● ●

class RAG(dspy.Module):
    def __init__(self, num_docs=5):
        self.num_docs = num_docs
        self.respond = dspy.Predict('context, question -> response')
        self.retrieve = ...

    def forward(self, question):
        context = retrieve(question, k=self.num_docs)
        return self.respond(context=context, question=question)

tp = dspy.MIPROv2(metric=dspy.evaluate.SemanticF1(decompositional=True))
optimized_rag = tp.compile(RAG(), trainset=trainset, max_bootstrapped_demos=2, max_labeled_demos=2)
```



# DSPy

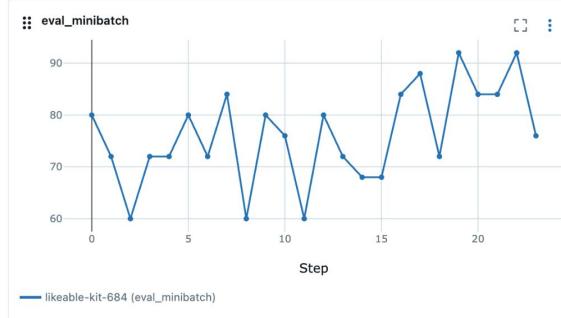
MLflowではmlflow.dspy.autolog()だけで最適化経過や指標のロギング、モジュール内部のトレーシングやStateの保存が可能。

最適化の成果物は  
Child Runとして保存

DSPy [Provide Feedback](#) [Add Description](#)

Runs	Evaluation	Experimental	Traces
<input type="checkbox"/> Run Name	Created	Data:	
<input type="checkbox"/> likeable-kit-684	5 minutes ago	-	
<input type="checkbox"/> thoughtful-goat-294	21 minutes ago	-	
<input type="checkbox"/> blushing-carp-334	2 days ago	-	
<input type="checkbox"/> eval_full_1	2 days ago	-	
<input type="checkbox"/> eval_minibatch...	2 days ago	-	
<input type="checkbox"/> eval_minibatch...	2 days ago	-	
<input type="checkbox"/> eval_minibatch...	2 days ago	-	
<input type="checkbox"/> eval_minibatch...	2 days ago	-	

評価指標の推移



内部で何が起こっているかを  
トレースで可視化

MLflow Trace UI [Learn More](#)

Search

Task name

- ChainOfThought.forward 0.01s
- Predict.forward 0.01s
  - ChatAdapter.format 0.01s
  - LM.\_\_call\_\_ 0.01s
  - ChatAdapter.parse 0.01s

Chat Inputs / Outputs Attributes Events

Messages

System

Your input fields are:

- tokens (list[str]): tokenized text

Your output fields are:

- reasoning (str)
- extracted\_people (list[str]): all tokens referring to a tokenized text

All interactions will be structured in the following way, with the

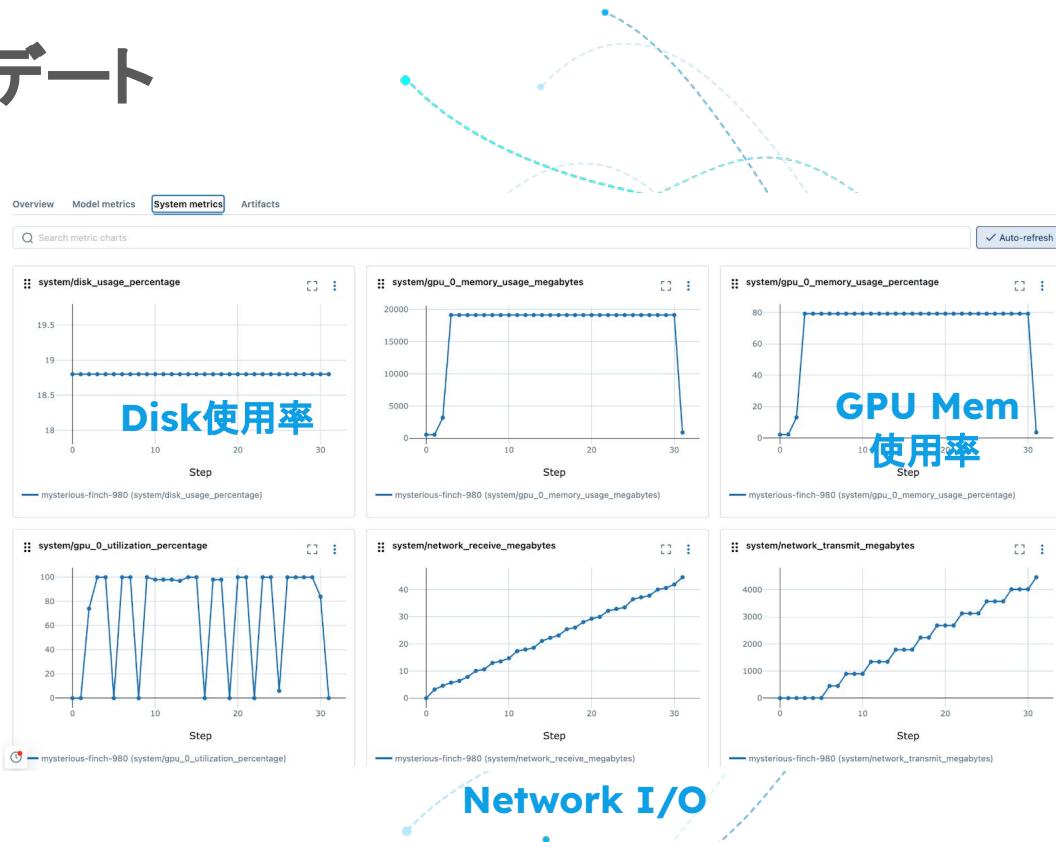
Show timeline

User

The screenshot shows the MLflow Trace UI interface. At the top, there's a search bar and a 'Task name' dropdown menu. The main area displays a call tree for a 'ChainOfThought.forward' task, which then branches into 'Predict.forward', 'ChatAdapter.format', 'LM.\_\_call\_\_', and 'ChatAdapter.parse'. Below the call tree, there's a 'Messages' section with a 'System' header. It lists input fields ('tokens') and output fields ('reasoning', 'extracted\_people'). A 'Show timeline' button is at the bottom left, and a 'User' section is at the bottom right.

# 深層学習向けの UXアップデート

- System Metricsのロギング
- チャートUIの拡充
- 大規模ロギング
- Pytorch/Keras自動ログ
- PEFT (LoRA) サポート



# 深層学習向けの UXアップデート

- System Metricsのロギング
- チャートUIの拡充
- 大規模ロギング
- Pytorch/Keras自動ログ
- PEFT (LoRA) サポート

Epochごとのラン比較



# 深層学習向けの UXアップデート

- System Metricsのロギング
- チャートUIの拡充
- 大規模ロギング
- Pytorch/Keras自動ログ
- PEFT (LoRA) サポート



## Parallel Runs

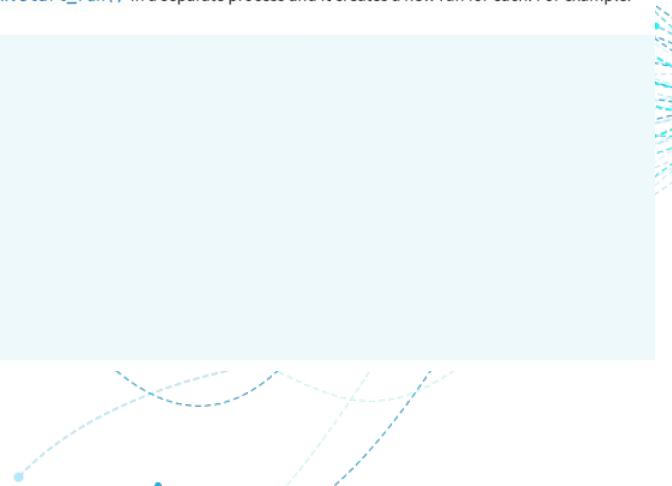
MLflow also supports running multiple runs in parallel using [multiprocessing](#) or multi threading.

Multi-processing is straightforward: just call `mlflow.start_run()` in a separate process and it creates a new run for each. For example:

```
import mlflow
import multiprocessing as mp

def train_model(params):
    with mlflow.start_run():
        mlflow.log_param("p", params)
        ...

if __name__ == "__main__":
    params = [0.01, 0.02, ...]
    pool = mp.Pool(processes=4)
    pool.map(train_model, params)
```



# 深層学習向けの UX アップデート

- System Metrics のロギング
- チャート UI の拡充
- 大規模ロギング
- チェックポイント自動ログ
- PEFT (LoRA) サポート

Experiments > /Repos/puneet.jain@dataricks.com/deeplearning\_with\_mlflow/train >  
mysterious-finch-980 Provide feedback □

Overview Model metrics System metrics Artifacts

checkpoints/epoch\_3/checkpoint\_metrics.json 31B  
Path: dbfs:/databricks/mlflow-tracking/6eedabb2f2964c38b5c97dc70ddde16/d5826dd21259474194b9c8b8a5c54afj/artifacts/checkpoints/epoch\_3/checkpoint\_metrics.json

```
{  
    "Val_Loss": 0.300042986869812,  
    "Val_Acc": 0.9052734375,  
    "Val_F1_Score": 0,  
    "Val_Precision": 0,  
    "Val_Recall": 0,  
    "Train_Loss": 0.33806294202804565,  
    "Train_Acc": 0.8861607313156128,  
    "Train_F1_Score": 0,  
    "Train_Precision": 0,  
    "Train_Recall": 0,  
    "epoch": 3,  
    "global_step": 84  
}
```

mlflow.pytorch.autolog() で自動的にチェックポイントと評価指標が保存されます

# 深層学習向けの UXアップデート

- System Metricsのロギング
- チャートUIの拡充
- Bulkロギング
- Pytorch/Keras自動ログ
- PEFT (LoRA) サポート

チュートリアルは[こちら](#)から

```
import mlflow

model_id = "meta-llama/Llama-3.2-3B-Instruct"
base_model = AutoModelForCausalLM.from_pretrained(model_id)
tokenizer = AutoTokenizer.from_pretrained(model_id)

peft_config = LoraConfig(...)
peft_model = get_peft_model(base_model, peft_config)

with mlflow.start_run():
    # Your training code here
    ...

    # Log the PEFT model
    model_info = mlflow.transformers.log_model(
        transformers_model={
            "model": peft_model,
            "tokenizer": tokenizer,
        },
        artifact_path="peft_model",
    )
```



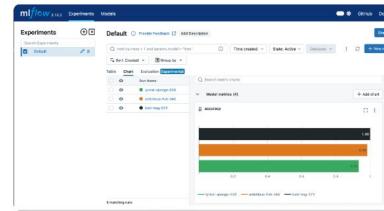
MLflowの1番(?)の欠点

Trackingサーバーを立てるのが面倒 ...

# マネージドサービスが次々と！

- **SageMaker**
- **Nebius**
- **Databricks**

- 2024年6月にリリース
- SageMakerのデプロイと連携
- 新しいUnified Studioにも
- こちらの[記事](#)で試しています



## Track experiments from anywhere

ML experiments are performed in diverse environments, including local notebooks, IDEs, cloud-based training code, or managed IDEs in [Amazon SageMaker Studio](#). With SageMaker AI and MLflow, you can use your preferred environment to train models, track your experiments in MLflow, and launch the MLflow UI directly or through SageMaker Studio for analysis.

## Collaborate on model experimentation

Effective team collaboration is essential for successful data science projects. SageMaker Studio allows you to manage and access the MLflow Tracking Servers and experiments, enabling team members to share information and ensure consistent experiment results, making collaboration easier.

# マネージドサービスが次々と

- SageMaker
- Nebius
- Databricks

- 2024年9月にリリース
- Nvidia出資のGPUクラウド
- LLMのFine-tuningがNebiusでのMLflowの主な使い方

The screenshot shows the Nebius website with a navigation bar at the top featuring links for AI Cloud, AI Studio, Solutions, Why Nebius, Pricing, Docs, Resources, Company, and For investors. On the right, there are buttons for Contact sales, Log in to AI Studio, and Log in to AI Cloud. A decorative graphic of overlapping colored layers (purple, blue, green) is on the right side.

**PREVIEW**

## Managed Service for MLflow

A fully managed service for an industry-leading tool for managing the machine learning lifecycle.

*The service is provided free of charge and is at the [Preview](#) stage.*

[Get started](#) [Documentation](#)

 **Zero infrastructure maintenance**  
Nebius offers MLflow, an industry-leading tool for optimizing the ML lifecycle, as a fully managed and ready-to-work cloud solution. This enables you to deliver production-ready models faster without having to worry about server provisioning.

 **Transparent ML pipeline**  
MLflow collects and organizes the metadata of your model training, making your ML pipeline more transparent and visible. That allows your ML team to control the progress and apply relevant changes with the highest level of precision.

 **Improved collaboration**  
Developing ML models creates a huge amount of information and assets that must be accessible to various stakeholders. MLflow provides efficient tools for organizing and easily sharing them across the team.



# マネージドサービスが次々と

- SageMaker
- Nebius
- **Databricks**

マネージドMLflowは元々無料で使えたが、Sign-upがより簡単になり誰でもチームでも無料でManaged MLflowが使えるように！

Googleアカウントやメールアドレスのみで登録可能！



## Sign up

Use your work email for the best experience and up to \$400 of credits. [Terms apply](#)

Continue with Google

Continue with Microsoft

or

Email

By continuing, you agree to Databricks' [Terms of Service](#), [Privacy Notice](#) and corporate email sharing terms [\(1\)](#)

[Continue with email](#)

# 3 Billion Devices Teams run MLflow



**mlflow™**



Azure Machine Learning



**databricks**



Amazon SageMaker



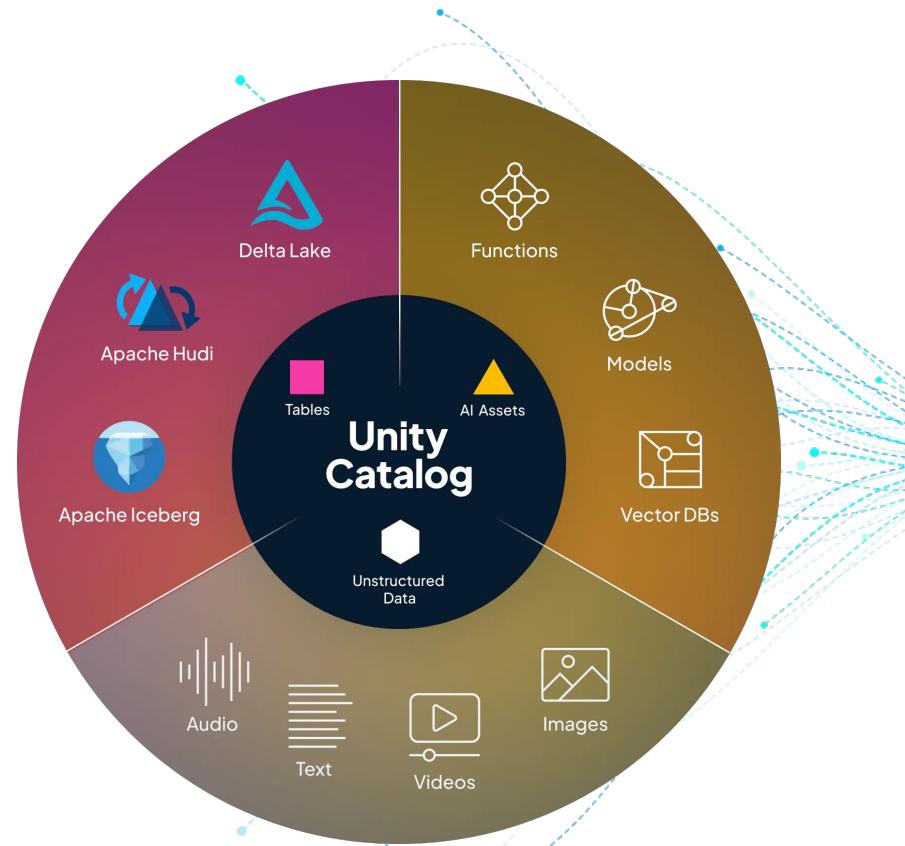
Google Kubernetes Engine

**NEBIUS**

**mlflow™**

# Unity Catalog AI

- Unity CatalogがOSSに
- Unity CatalogをOSS MLflowの  
モデルレジストリとして利用できます！
- アクセス管理、ガバナンスなどなど



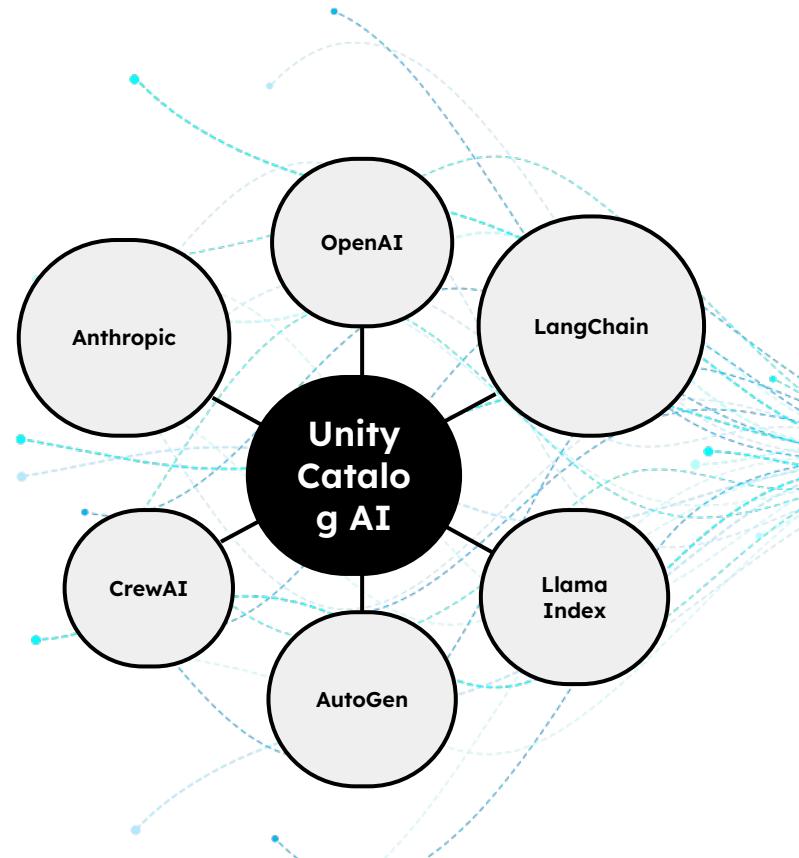
# Unity Catalog AI

- PythonやSQLの関数をUCに登録することで、サンドボックス内でツール実行できます。
- [Unity Catalog AI](#)パッケージでは、様々なライブラリとUCツールの連携が簡単に行えます。

```
import anthropic
from unitycatalog.ai.anthropic.toolkit import UCFFunctionToolkit

toolkit = UCFFunctionToolkit(function_names=[f"{{CATALOG}}.{{SCHEMA}}.{{name}}"])
tools = toolkit.tools

response = anthropic.Anthropic().messages.create(
    model="claude-3-5-sonnet-latest",
    max_tokens=1024,
    tools=tools,
    messages=[{
        "role": "user",
        "content": "What's the weather in Nome, AK and in Death Valley, CA?"
    }],
)
```



# プロンプトレジストリ (new!)

- つい先週リリース
- プロンプトをバージョン管理するための機能
- 私もClaude+MCP用のプロンプトを早速保存して使っています

The screenshot shows the mlflow UI with the 'Prompts' tab selected. A specific prompt named 'summarization-prompt' is displayed. The prompt details are as follows:

Version	Version 1 (baseline)	Version 2
Registered at:	2025-03-13 09:53:37	2025-03-13 09:56:24
Aliases:	Add	Add
Commit message:	Initial commit	Improvement
Metadata:	author: author@example.com	author: author@example.com

The 'Compare' tab is active, showing the differences between Version 1 and Version 2. The 'Preview' tab shows the prompt content:

```
Summarize content you are provided with in {{ num_sentences }} sentences.
```

The 'Sentences: {{ sentences }}' field is also visible.

The 'Comparing version 1 with version 2' section highlights the changes made in Version 2:

- Summarize You content are provided with in {{ num\_sentences }} sentences.
- Sentences: {{ sentences }}
- Your summary should:
  - Contain exactly {{ num\_sentences }} sentences
  - Include only the most important information
  - Be written in a neutral, objective tone
  - Maintain the same level of formality as the original text

# ドキュメントの刷新

- ドキュメントもプロダクトの一部として、頻繁に改善しています。
- サイドバーと検索ツールが変更されて、見たいページを探しやすくなりました。
- "Ask AI"という機能を使えば、ドキュメントやソースコードの情報を元に答えてくれます。

The screenshot shows the MLflow website interface. At the top, there's a navigation bar with links for GitHub, search, and account information. The main content area has two main sections:

- Experiment Tracking**: This section discusses the core feature of organizing assets and artifacts during LLM/GenAI projects. It includes a chart showing a bell-shaped curve and a screenshot of the MLflow UI showing various runs and metrics.
- Tracing / Observability**: This section explains how MLflow ensures observability by capturing LLM calls and other details. It includes a screenshot of the MLflow UI showing tracing data over time.

On the right side, there's a sidebar with links to other MLflow features like Model Packaging, Evaluation, and Model Serving, along with a "Why Choose MLflow?" section and a "How MLflow Solves LLM Ops Challenges" section.

# 2025年のMLflow

# MLflow in 2025 (前半)

フィードバックループ

アノテーションとデータセットの Built-inサポート

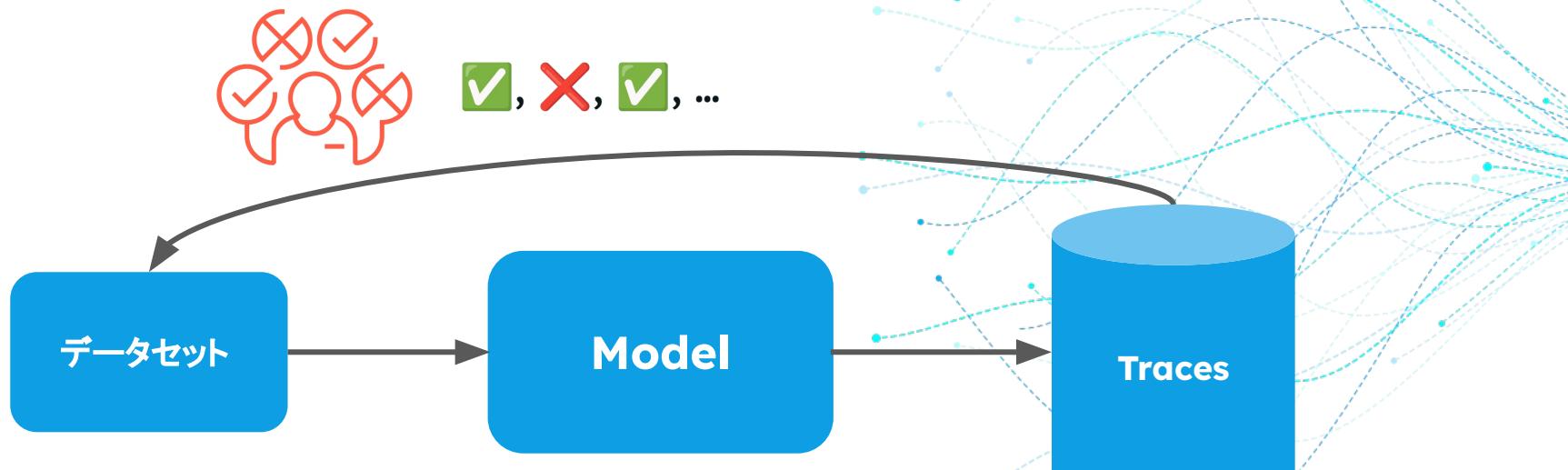
モデルを中心とした強固なリネージ

デプロイ管理

モデルレジストリを CI/CDと更に連携

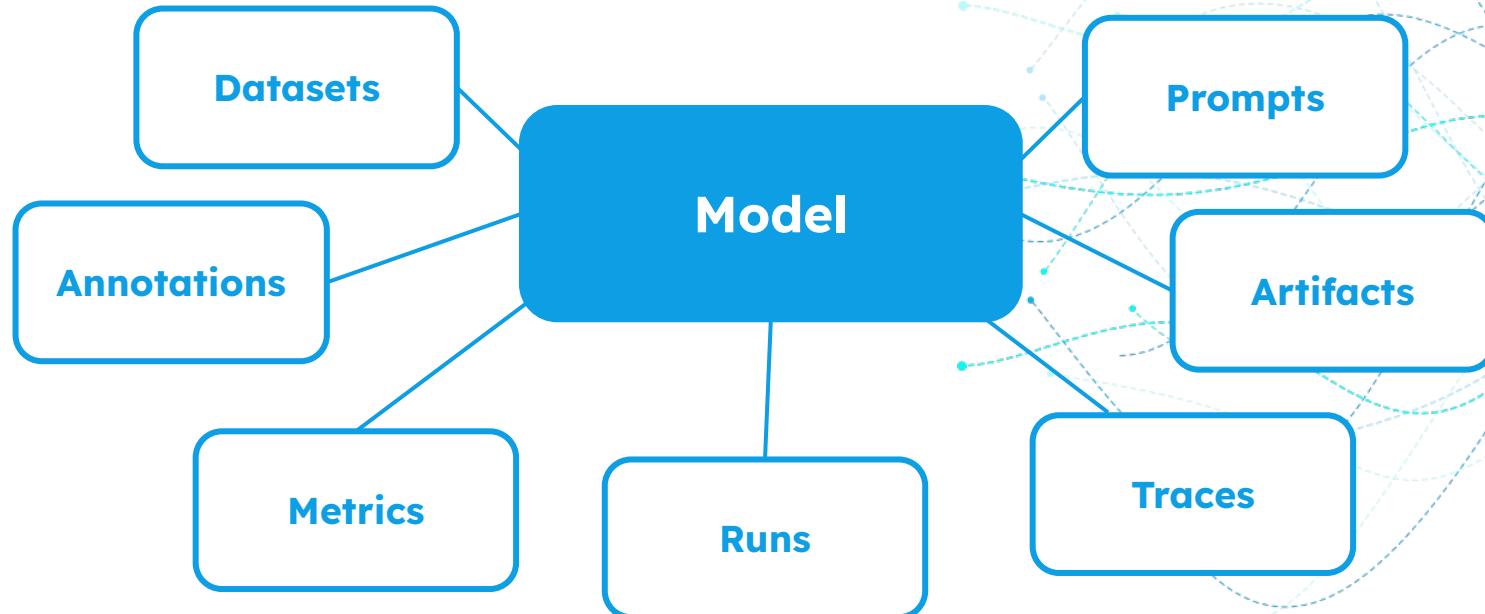
# MLflow 2025: モデルの質を高めるフィードバックループ

MLflowでデータセットの管理とアノテーションが可能になります！

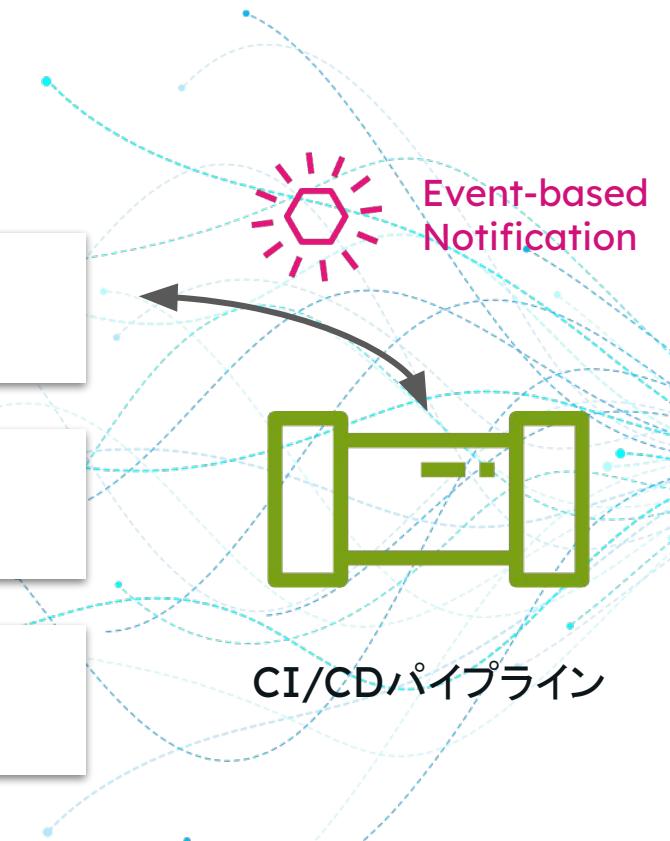


# MLflow 2025: より強固なリネージ

モデルを中心に、全てのリソースに対してリネージと検索を可能に。



# MLflow 2025: デプロイ管理



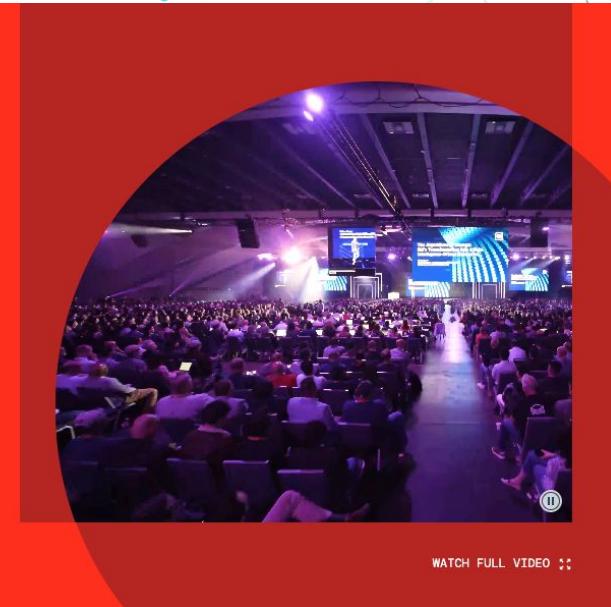
# MLflow in 2025

まだまだここでは紹介できないアップデートが ....  
続々は6月の **Data and AI Summit** にて！



## REGISTRATION NOW OPEN

Join 20,000 of your peers for 700+ sessions, keynotes and training at the world's largest data, analytics and AI conference

[REGISTER](#)

[WATCH FULL VIDEO](#)

# Thank you!



[mlflow-users.slack.com](https://mlflow-users.slack.com)



[github.com/mlflow/mlflow](https://github.com/mlflow/mlflow)



@MLflow (mlflow-org)



@MLflow



[lu.ma/mlflow](https://lu.ma/mlflow)

